Listing of Claims:

1. (Previously Presented) A method of accessing an Enterprise Java Bean (EJB) by a non-Java application within a computing environment, comprising:

a) making a call, by the non-Java application, to a client library, wherein the call includes input parameters;

b) invoking a function within the client library to construct an HTTP request corresponding to the input parameters of the call made from the non-Java application;

c) passing the HTTP request from the client library to an EjbServlet;

d) invoking a method on an EJB by the EjbServlent based upon the HTTP request;

e) returning information from the invoked method from the EJB to the EjbServlet;

f) decoding the returned information into an HTTP response string by the EjbServlet;

g) transmitting the HTTP response from the EjbServlet to the client library; and

h) parsing and converting the HTTP response by the client library into return information compatible with the non-Java application and then passing the return information from the client library to the non-Java application.

2. (Original) The method of claim 1 wherein the invoked method is a chained method.

3. (Original) The method of claim 1 wherein the invoking a method on the EJB by the EjbServlet further comprises passing one or more input parameters by the EjbServlet to the EJB.

4. (Original) The method of claim 3 wherein the input parameters compirse one or more input objects constructed by the EjbServlet based on the HTTP request.

5. (Original) The method of claim 1 wherein the returning information from the invoked method from the EJB to the EjbServlet further comprises the EJB constructing a return object based upon the information form the invoked method and passing the return object to the EjbServlet.

6. (Original) The method of claim 5 wherein the return object is a chained object.

7. (Original) The method of claim 5 wherein the decoding the returned information into an HTTP response string by the EjbServlet further comprises decoding the return object into an HTTP response string by the EjbServlet.

8. (Original) The method of claim 1 wherein the HTTP request and the HTTP response each comprise a series of bytes representing HTTP-specific control information or text strings.

9. (Original) The method of claim 8 wherein the HTTP request and the HTTP response are passed between the client library and the EjbServlet via an HTTP protocol.

10. (Original) The method of claim 9 wherein the HTTP protocol enables the client library and EjbServlet to communicate across a distributed computing environment.

11. (Original) The method of claim 1 wherein the non-Java application is based on a programming environment capable of calling external library functions via the C calling convention.

12. (Original) The method of claim 1 further comprising the non-Java application allocating buffers to hold calling input parameters sent to the client library and return information received from the client library.

13. (Original) The method of claim 1 wherein the client library is a linkable library.

14. (Original) The method of claim 1 wherein the calls between the client library and the non-Java application are based upon the C language calling convention.

15. (Original) The method of claim 1 further comprising the client library converting any numeric input parameter in the calling input parameters into a corresponding text representation in the HTTP request.

16. (Original) The method of claim 12 further comprising the client library extracting the return information from the HTTP response sent by the EjbServlet and placing the return information into the buffers provided by the non-Java calling application.

17. (Original) The method of claim 16 further comprising the client library converting any text-represented numeric value extracted from the HTTP response into a corresponding numeric form thereof.

18. (Original) The method of claim 1 wherein the EjbServlet and the method invoked on the EJB is identified by a calling input parameter embedded in the HTTP request.

19. (Original) The method of claim 1 wherein the method is invoked via a remote method invocation (RMI) protocol.

20. (Original) The method of claim 19 wherein the RMI enables the EjbServlet and the EJB to communicate across a distributed computing environment.

21. (Original) The method of claim 12 wherein the return information is placed into two buffers.

22. (Original) The method of claim 21 wherein the buffers comprise a data buffer and a format buffer.

23. (Original) The method of claim 22 further comprising passing additional decoded return information wherein the information from the invoked method exceeds the data buffer capacity, the format buffer capacity, or both.

24. (Original) The method of claim 23 wherein the EjbServlet stores the remaining decoded EJB method call results in memory.

25. (Original) The method of claim 23 wherein the client library passes a return code to the non-Java application indicating that information from the invoked method remains in the EjbServlet.

26. (Original) The method of claim 23 wherein additional return data and format strings are passed until all of the decoded information from the invoked method is received by the non-Java application.

27. (Original) The method of claim 23 wherein the EjbServlet passes a key to the client library identifying any information from the invoked method remaining in the EjbServlet.

28. (Original) The method of claim 27 wherein the client library places the key in a session ID parameter.

29. (Original) The method of claim 28 wherein the client library provides the key to the non-Java application.

30. (Original) The method of claim 29 wherein non-Java application accesses the information from the invoked method remaining in the EjbServlet using the key.

31. (Original) The method of claim 1 further comprising invoking a logging function within the client library.

32. (Original) A computing system for accessing an EJB by a non-Java application comprising:

a) a non-Java application in communication with a client library;

b) a means for calling the client library from the non-Java application wherein said means for calling the client library is used to establish communication between the non-Java application and the client library;

c) an EjbServlet in communication with the client library wherein the client library comprises a function to take input parameter information from the call, embed the information into an HTTP request, and transfer the request to the EjbServlet;

d) a means for transferring information between the client library and the EjbServlet wherein said means for transferring it is used to establish communication between the EjbServlet and the client library via an HTTP protocol;

e) the EjbServlet configured to receive the HTTP request from the client library and invoke a corresponding method on an EJB; and

f) a remote method interface (RMI) for invoking methods and returning Java objects between the EjbServlet and the EJB.

33. (Previously Presented) The computer system of claim 32 wherein the EbjServlet constructs one or more input objects to invoke a method on the EJB.

34. (Previously Presented) The computer system of claim 32 wherein the EjbServlet receives a returned Java object from the EJB and converts the object to strings.

35. (Previously Presented) The computer system of claim 34 wherein the EjbServlet constructs an HTTP response containing the return strings.

36. (Previously Presented) The computer system of claim 35 wherein the EjbServlet transfers the HTTP response to the client library.

37. (Previously Presented) The computer system of claim 36 wherein the client library parses and converts the HTTP response into a return parameter string that is returned to the non-Java application, wherein the return parameter string is compatible with the non-Java application.